Tutorial and Advance Troubleshooting using Process Explorer

Process Explorer is one powerful tool which you can do lot of troubleshooting and Developers level debugging programs. According to Process Explorer help file

“Process Explorer is an advanced process management utility that picks up where Task Manager leaves off. It will show you detailed information about a process including its icon, command-line, full image path, memory statistics, user account, security attributes, and more. When you zoom in on a particular process you can list the DLLs it has loaded or the operating system resource handles it has open. A search capability enables you to track down a process that has a resource opened, such as a file, directory or Registry key, or to view the list of processes that have a DLL loaded.

The Process Explorer display consists of two sub-windows. The top always shows a list of the currently active processes, including the names of their owning accounts, whereas the information displayed in the bottom window, which you can close, depends on the mode that Process Explorer is in: if it is in handle mode you will see the handles that the process selected in the top window has opened; if Process Explorer is in DLL mode you will see the DLLs and memory-mapped files that the process has loaded.

Process Explorer also has a powerful search capability that will quickly show you which processes have particular handles opened or DLLs loaded. The unique capabilities of Process Explorer make it useful for tracking down DLL-version problems or handle leaks, and provide insight into the way Windows and applications work. “

There are few basics you need to know before getting into Advance troubleshooting technique.

Process and Threads

A Windows process is essentially container that hosts the execution of an executable image file. It is represented with a kernel process object and Windows uses the process object and its associated data structures to store and track information about the image’s execution. For example, a process has a virtual address space that holds the process’s private and shared data and into which the executable image and its associated DLLs are mapped. Windows records the process’s use of resources for accounting and query by diagnostic tools and it registers the process’s references to operating system objects in the process’s handle table. Processes operate with a security context, called a token, that identifies the user account, account groups, and privileges assigned to the process.

Finally, a process includes one or more threads that actually execute the code in the process (technically, processes don’t run, threads do) and that are represented with kernel thread objects. There are several reasons applications create threads in addition to their default initial thread: processes with a user interface typically create threads to execute work so that the main thread remains responsive to user input and windowing commands; applications that want to take advantage of multiple processors for scalability or that want to continue executing while threads are tied up waiting for synchronous I/O operations to complete also benefit from multiple threads

Handles

Usually, processes need to access OS resources. Disk read/write, graphics/text to screen, mouse, more/less memory, etc. Any call to OS resources requires the OS to schedule and allocate the resource to the process. With many processes requesting OS resources, the OS needs an orderly mechanism to allocate them: thus, handles

When a process is initialized, the system allocates a handle table for it. This handle table is used only for kernel objects, not for User objects or GDI objects. When a process first initializes, its handle table is empty. Then when a thread in the process calls a function that creates a kernel object, such as CreateFileMapping, the kernel allocates a block of memory for the object and initializes it; the kernel then scans the process's handle table for an empty entry

Virtual Memory Types

Process Committed: contents are backed by a file on disk (data file, image or paging file)

Address space breakdown

Committed:

· *Shareable* (e.g. EXE, DLL, shared memory, other memory mapped files)

· *Private* (e.g. process heap)

Uncommitted:

· *Reserved* (not yet committed)

· *Free* (not yet defined)

Pages in a process virtual address space are free, reserved, or committed. Applications can first reserve address space and then commit pages in that address space. Or they can reserve and commit in the same function call. Reserved address space is simply a way for a thread to reserve a range of virtual addresses for future use. Attempting to access reserved memory results in an access violation because the page isn't mapped to any storage that can resolve the reference.

Committed pages are pages that, when accessed, ultimately translate to valid pages in physical memory. Committed pages are either private and not shareable or mapped to a view of a section (which might or might not be mapped by other processes). Sections are described in two upcoming sections, "Shared Memory and Mapped Files" and "Section Objects."

If the pages are private to the process and have never been accessed before, they are created at the time of first access as zero-initialized pages (or demand zero). Private committed pages can later be automatically written to the paging file by the operating system if memory demands dictate. Committed pages that are private are inaccessible to any other process unless they're accessed using cross-process memory functions.

If committed pages are mapped to a portion of a mapped file, they might need to be brought in from disk when accessed unless they've already been read earlier, either by the process

accessing the page or by another process that had the same file mapped and had previously accessed the page, or if they've been prefetched by the system.

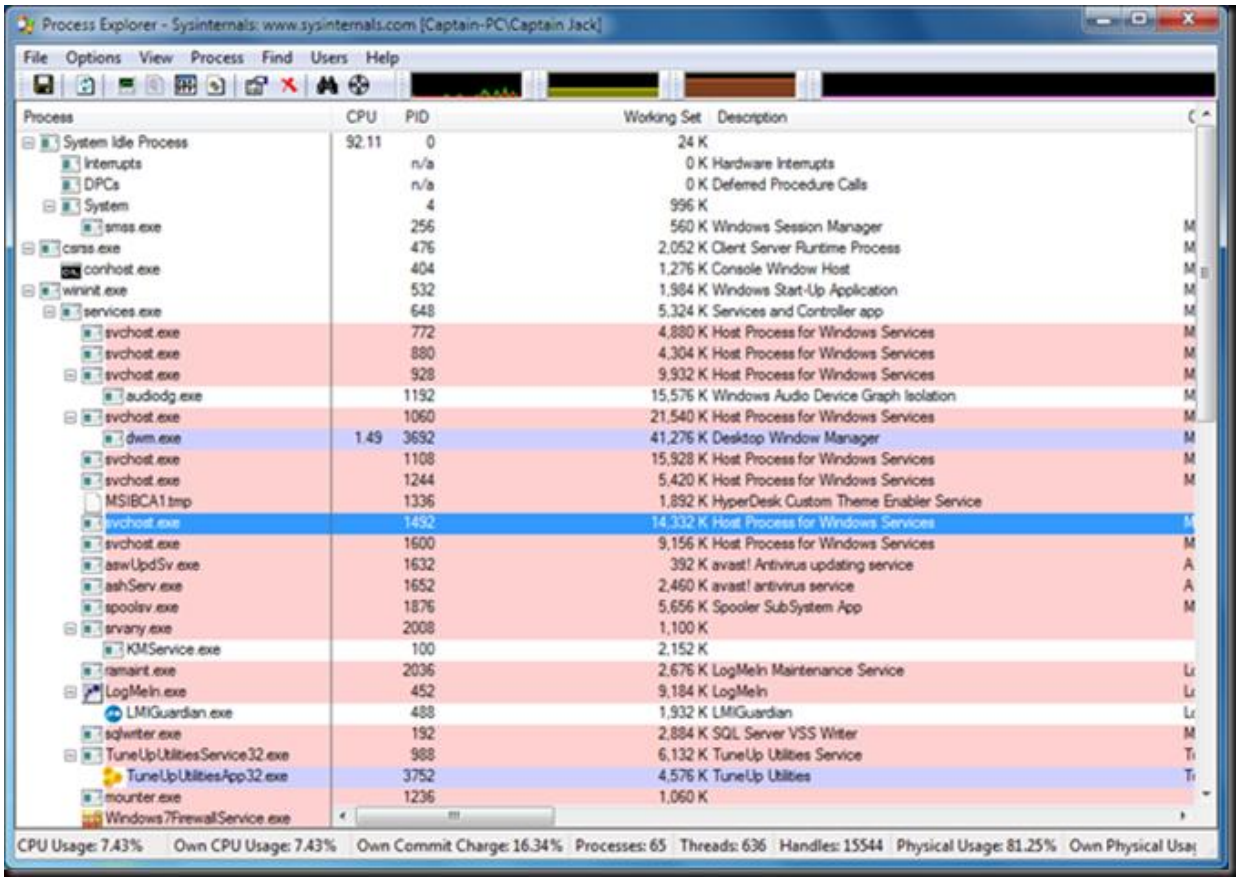
More description is mentioned in the article
http://members.shaw.ca/bsanders/WindowsGeneralWeb/RAMVirtualMemoryPageFileEtc.htm

I think these are few basics we need to know while using Process Explorer. I won't be mentioning all the options about Process Explorer, but few options that would be helpful in troubleshooting issues with Windows.

Super Task Manager

Most of them call Process Explorer a super Task Manager because of the features it has. One main reason is you can break down a process to threads and handles etc. we can really dig down deep even to Kernel level.

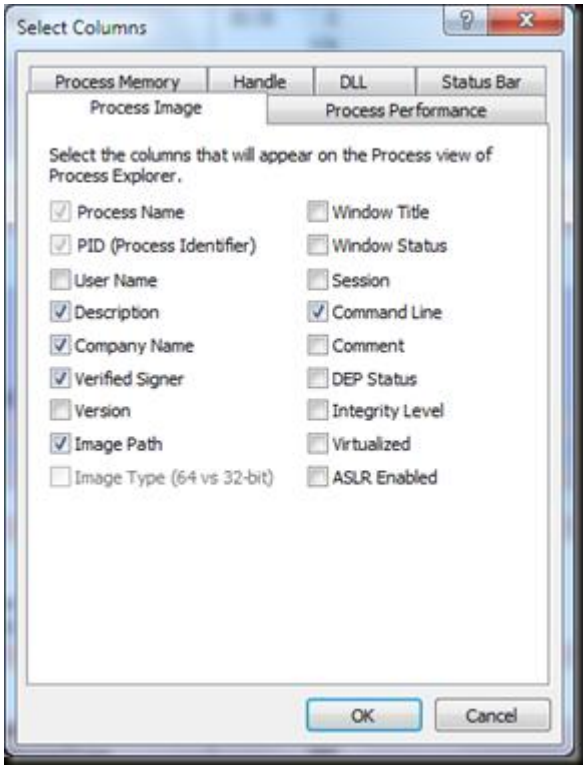
This is what it would look like:


Process Explorer - Sysinternals: www.sysinternals.com [Captain-PC\Captain Jack]
File Options View Process Find Users Help
Process CPU PID Working Set Description
System Idle Process 92.11 0 24 K
Interrupts n/a 0 K Hardware Interrupts
DPCs n/a 0 K Deferred Procedure Calls
System 4 996 K
smss.exe 256 560 K Windows Session Manager
csrss.exe 476 2,052 K Client Server Runtime Process
conhost.exe 404 1,276 K Console Window Host
wininit.exe 532 1,984 K Windows Start-Up Application
services.exe 648 5,324 K Services and Controller app
svchost.exe 772 4,880 K Host Process for Windows Services
svchost.exe 880 4,304 K Host Process for Windows Services
svchost.exe 928 9,932 K Host Process for Windows Services
audiodg.exe 1192 15,576 K Windows Audio Device Graph Isolation
svchost.exe 1060 21,540 K Host Process for Windows Services
dwm.exe 1.49 3692 41,276 K Desktop Window Manager
svchost.exe 1108 15,928 K Host Process for Windows Services
svchost.exe 1244 5,420 K Host Process for Windows Services
MSIBCA1.tmp 1336 1,892 K HyperDesk Custom Theme Enabler Service
svchost.exe 1492 14,332 K Host Process for Windows Services
svchost.exe 1600 9,156 K Host Process for Windows Services
aswUpdSv.exe 1632 392 K avast! Antivirus updating service
ashServ.exe 1652 2,460 K avast! antivirus service
spoolsv.exe 1876 5,656 K Spooler SubSystem App
srvany.exe 2008 1,100 K
KMService.exe 100 2,152 K
ramaint.exe 2036 2,676 K LogMeIn Maintenance Service
LogMeIn.exe 452 9,184 K LogMeIn
LMIGuardian.exe 488 1,932 K LMIGuardian
sqlwriter.exe 192 2,884 K SQL Server VSS Writer
TuneUpUtilitiesService32.exe 988 6,132 K TuneUp Utilities Service
TuneUpUtilitiesApp32.exe 3752 4,576 K TuneUp Utilities
mounter.exe 1236 1,060 K
Windows7FirewallService.exe
CPU Usage: 7.43% Own CPU Usage: 7.43% Own Commit Charge: 16.34% Processes: 65 Threads: 636 Handles: 15544 Physical Usage: 81.25%

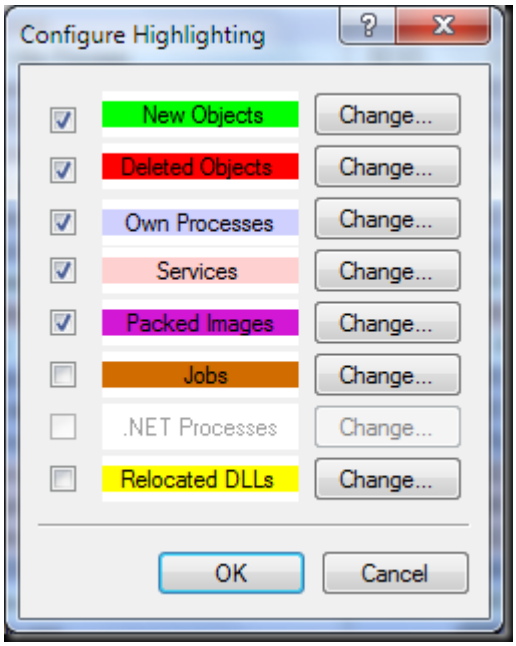
Here you can see the Process can be expanded and we can see what threads are running beneath. This comes helpful when you're troubleshooting Memory or Hung process or even Malicious software removal.

When you hover over one of the Processes it would show you the Path of the File from where it's running which you don't have in Task Manager.

In Process Explorer you have various columns. You could access them from the View Menu.



You can pick the one you want according to what you are troubleshooting. For example if you're Troubleshooting Memory then you can pick **Page Faults, Private Bytes, Private Byte History and Working set size.** The best part about this is you could save a view you want to. Just go to View you can click on **Save Column Set** you can name it and Later one if you want this view again go to View menu and select **Load Column Set.**



Process View Options

In Process Explorer you can find lot of color Highlighted each represents different purpose. You can change them from Option > Configure Highlighting

Configure Highlighting: select this menu item under the Options menu to open a dialog box that allows you to configure highlight colors used in the Process View and the DLL view.

Highlight Services: on Windows NT and higher this option has Process Explorer show processes that are running Win32 services in the service process highlight color. The Services tab of the process properties dialog shows the list of services running within a process.

Highlight Jobs: on Windows 2000 and higher choose this option to have Process Explorer show processes that are part of a Win32 Job in the Job object highlight color. Jobs group processes together so that they can be managed as a single item and are used by the Runas command, for example. Use the Job tab of the process properties dialog to see the list of processes running in the same job as the selected process and to see job limits that have been applied to the job.

Highlight .NET Processes: this option appears on Windows NT-based systems that have the .NET Framework installed. When the option is checked managed applications (those that use the .NET Framework) are highlighted in the .NET process highlight color.

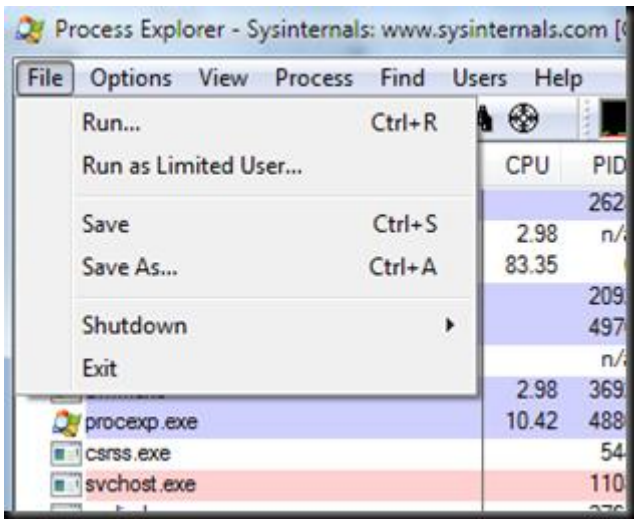
Highlight Own Processes: on Windows NT and higher checking this option results in Process Explorer showing in the own-process highlight color the processes that are running in the same user account as Process Explorer.

Highlight Packed Images: malware, including viruses, spyware, and adware is often stored in a packed encrypted form on disk in order to attempt to hide the code it contains from antispyware and antivirus.

Show Fractional CPU: when this option is selected Process Explorer shows CPU usage to two decimal places. This can be useful to identify processes that would otherwise appear idle, but that are performing background processing.

Show New Processes: when enabled Process Explorer scrolls the Process view to bring into view new processes.

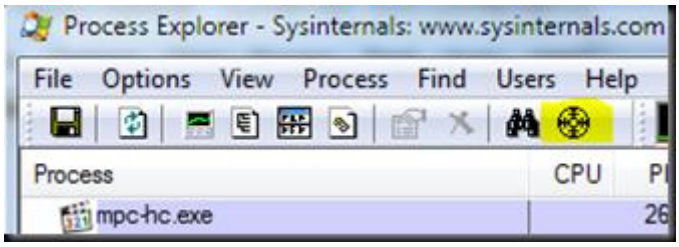
Another very useful option is Under **File** Menu



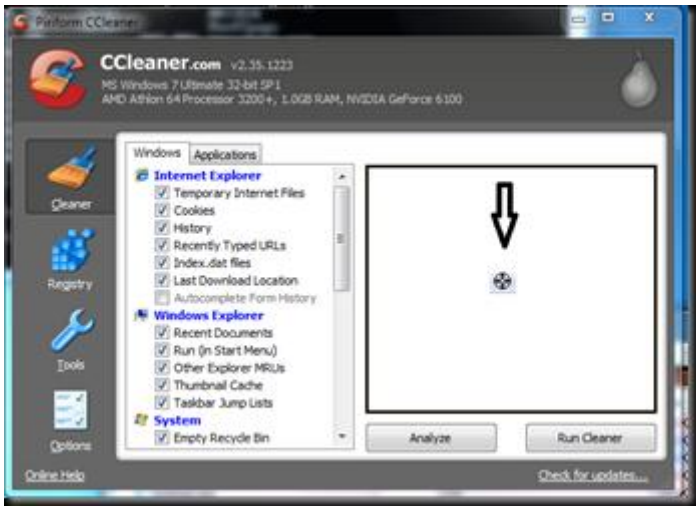
Run – It's the regular Start Run option

Run as Limited User – Its open an Application in Limited User Restriction. It comes handy when you want to open a Application in Limited User rights and see if that works or Open an Internet Explorer so it will be Safe to browser since it not running with Administrator rights.

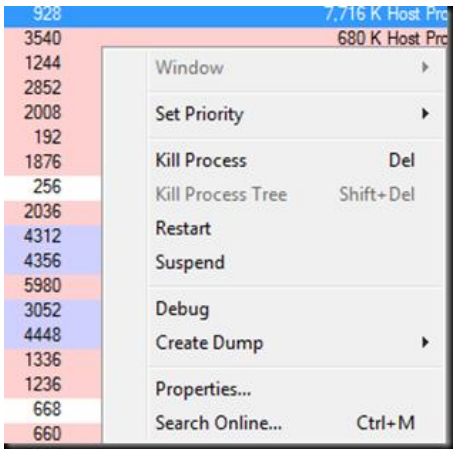
Window Finder Tool – Its another useful option in Process Explorer where it will find you which Process is using a Particular Window. For example you get a sudden Window popup and you want to know which Process is doing it you can use Window Finder Tool



Just Click on the Windows Finder Tool and drag it to the Window you want it to find the Window Process



The Process Context Menu



When you have a process selected the items in the Process menu become active. You can access the same menu items by right-clicking on a process. The items enable you to do the following:

Bring to Front: select this option to bring any windows owned by the selected process to the foreground.

Set Priority: you can change the base priority of a process with this submenu. When you change the base priority of a process the system adjusts the priorities of threads within the process so that they remain at the same relative priority with respect to the new base priority.

Set Affinity: on systems with multiple CPUs this menu item lets you bind the threads of a process to particular CPUs.

Debug: choosing this menu item launches the debugger registered in HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\AeDebug with the selected process as the command-line argument.

Launch Depends: if Process Explorer finds the Dependency Walker tool (see http://www.dependencywalker.com) with the selected process as the argument. The Dependency Walker tool shows static DLL dependencies.

Kill: this item terminates a process with the Terminate Process API. Note that a process terminated in this way is not warned of its termination and therefore does not write unsaved data it may have.

Kill Process Tree: if the process pane is in the process tree sorting mode this menu item is available and allows you to kill a process and all of its descendants.

Suspend: if you want a process to become temporarily inactive, so that a system resource such as network, CPU or disk, becomes available for other processes, you can suspend the

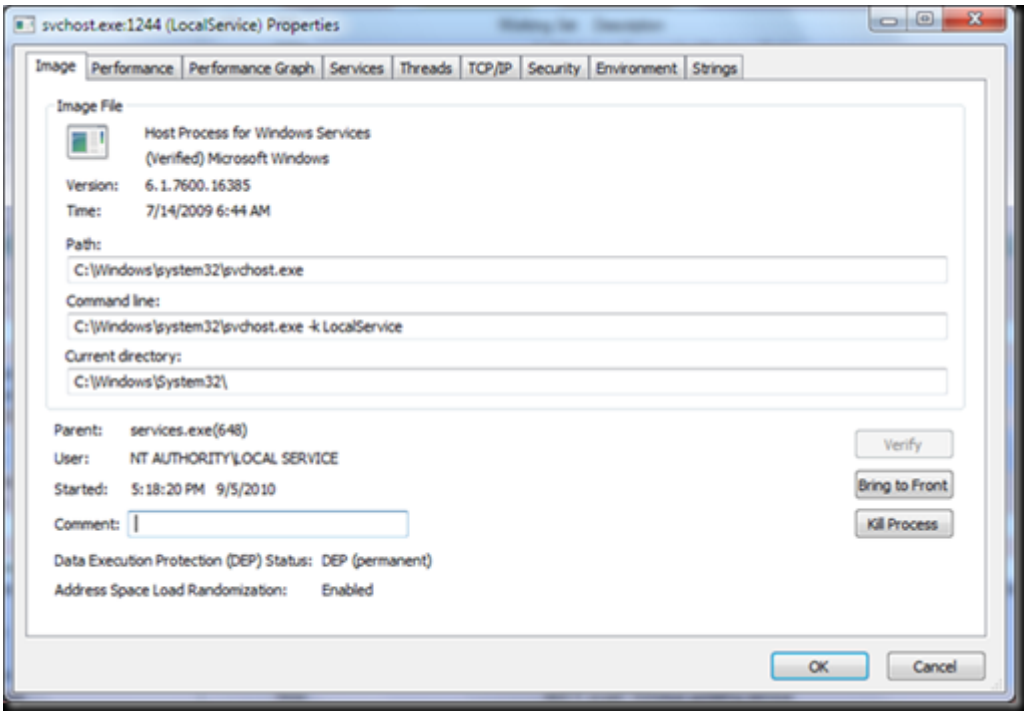
process. Suspended processes show in a dark grey color. To resume a suspended process chose the Resume item from the process context menu.

Restart: when you select this item Process Explorer terminates the highlighted process and starts the same image using the same command-line arguments. Note that the new instance may fail to run or behave differently if the original process ran in a different user account or had a different environment.

Properties: this selection opens a property dialog that shows you more information about a process.

Search Online: selecting this entry will result in Process Explorer launching the system's configured Internet browser and initiating an Internet search for the selected process' name.

Process Properties



You can view additional details for a process by double-clicking on it, or by selecting it and using the Process|Properties menu item or the properties toolbar button. On Windows 9x systems the dialog shows version information for the process image, the full path of the process image file, and the command-line used to launch the process. On Windows NT and higher there are several tabs in the dialog, described below. Any dynamic data, such as performance information, updates at the refresh date currently selected for Process Explorer. You can manually refresh dynamic information by typing F5 in a page.

Image:

This page shows version information extracted from the process' image file, the full path of the image file and the command-line that launched the process. It also shows the current directory of the process, the user account in which the process is running, the name of the process' parent process, and the time at which the process started execution.

Process Explorer checks for whether or not an image has been digitally signed by a certificate root authority trusted by the computer and displays the status of the check, which is either "Trusted" (signed), "Unsigned", or "Not Verified" (signature has not been checked). You can press the Verify button to have Process Explorer check the signature of an image that has not been verified. Note that the verification operation can result in Process Explorer contacting web sites to check for certificate validity. See the Verify Image Signatures option.

Enter a comment for a process in the Comment field. Comments are visible in the process view in the Comment column, or if you do not have the comment column selected, in the tool tip that displays when you hover the mouse over a process. Comments apply to all processes with the same path and are remembered from execution to execution.

On systems that support Data Execution Protection (DEP), Process Explorer shows the DEP status of the selected process as either "on" or "off". Software DEP is currently supported by Windows XP SP2 and higher on 32-bit x86 systems whereas hardware DEP is available only on 64-bit versions of Windows. You can also view DEP status by adding the corresponding DEP Status column to the process view.

Malware, including viruses, spyware, and adware is often stored in a packed encrypted form on disk in order to attempt to hide the code it contains from antispyware and antivirus. Process Explorer uses a heuristic to determine if an image is packed and if it is changes the text above the full path display field to include "(Image is probably packed)".

Performance:

Memory and CPU performance data displays on this page, including physical and virtual memory, and CPU usage. The data refreshes at the same interval that the main display does.

Performance Graph:

A history of a process' CPU usage and its private bytes allocation shows as in Task Manager-like graphs on this page. Red in the CPU usage graph indicates CPU usage in kernel-mode whereas green is the sum of kernel-mode and user-mode execution. Private Bytes represents the amount of private virtual memory a process has allocated and is the value that will rise of a process exhibiting a memory leak bug. Note that while the System Information performance graphs update while Process Explorer is minimized to the tray, these graphs do not. The private bytes usage graphs are scaled against the peak amount of private bytes the process has allocated; if the peak grows the graphs recalculate their scales. In the I/O graph the blue line indicates total I/O traffic, which is the sum of all process I/O reads and writes, between refreshes and the pink line shows write traffic. The I/O graph is scaled against the peak I/O traffic the process has generated since the start of monitoring.

Moving the mouse over part of a graph results in the time of the corresponding data point being shown in the graph as a popup either on the far left or right.

Threads:

The list of the threads running in the process shows on this tab. The thread list shows start address information that's provided by the Windows symbol engine. If you want to see accurate names for start addresses then follow the directions for configuring symbols.

The Module button on the threads page launches Explorer's file properties dialog box for the image file that contains the start address of the currently selected thread. The Stack button

shows the current stack of the selected thread. Stack information is unreliable unless symbol files are available for process and DLLs referenced in the stack.

Use the Kill button to terminate a thread. Note that terminating a thread may lead to a crash or erratic behavior of the process.

Use the Suspend button to suspend a thread. Note that suspending threads may cause its process to stop executing.

TCP/IP:

Any active TCP and UDP endpoints owned by the process are shown on this page.

On Windows XP SP2 and higher this page includes a Stack button that opens a dialog that shows the stack of the thread that opened the selected endpoint at the time of the open. This is useful for identifying the purpose of endpoints in the System process and Svchost processes because the stack will include the name of the driver or service that is responsible for the endpoint.

Security:

Process Explorer reports the list of groups and privileges listed in the security token of the process on this page. Privileges shown in grey are disabled. The permissions button opens a permissions editor that shows the access permissions assigned to the process.

Job:

This tab is present only for processes that are part of a Win32 Job. The Job page shows the list of processes that are part of the same job and the limits that are applied to the job.

.NET Assemblies:

This tab is present on Windows Vista and higher when Process Explorer runs with administrative rights and only for managed processes, which are those that use the .NET Framework. AppDomains and the assemblies loaded in each are displayed in a tree view.

.NET Performance:

This tab is present only for managed processes, which are those that use the .NET Framework. The AppDomains present in the process show, as well the available .NET performance counter objects. Select a .NET performance object to see the values of the object's counters. The counters update at the currently selected refresh interval and you can type F5 to manually refresh.

Services:

This tab is present only for processes that are executing Win32 services, and lists the services running within the process. Process Explorer shows a service's name and display name, and on Windows 2000 and higher, if available, the service's description. The permissions button opens a permissions editor that shows the access permissions assigned to the service.

Environment:

The environment variables associated with the process show on this page.

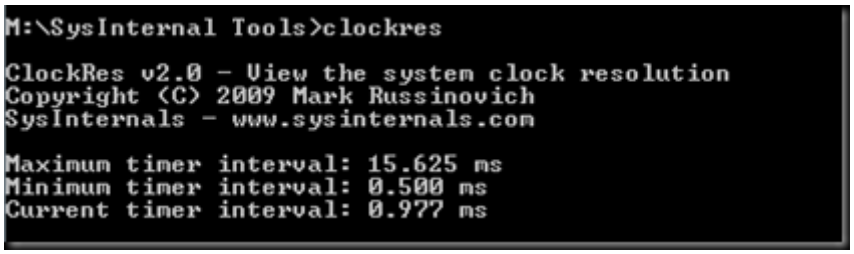
Strings:

All printable strings of at least 3 characters in length display on this page. Image strings are read from the process image file on disk whereas Memory strings are read from the image's in-memory storage. Memory strings may be different than on-disk strings when an image uses a decompresses or decrypts when it loads into memory.

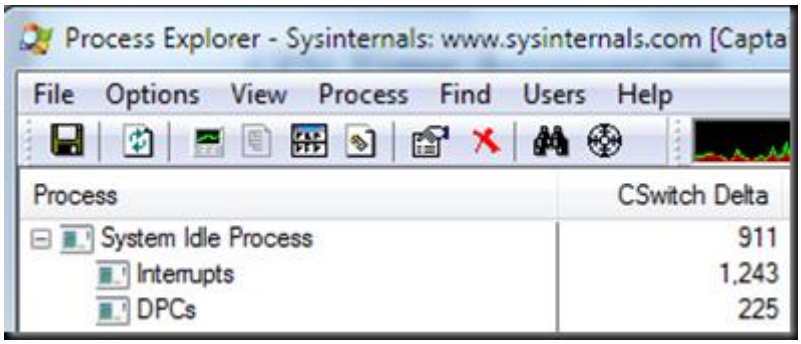
CPU Time Accounting

Windows Time Accounting is on the Motherboard called Clock Motherboard time. Every 15 milliseconds (for 32 bit OS)it will interrupt a process.

Using a Tool from SysInternals called **Clockres** you can find out what is yours


M:\SysInternal Tools>clockres
ClockRes v2.0 - View the system clock resolution
Copyright (C) 2009 Mark Russinovich
SysInternals - www.sysinternals.com
Maximum timer interval: 15.625 ms
Minimum timer interval: 0.500 ms
Current timer interval: 0.977 ms

So in every 15 milliseconds the Clock interrupts a Process and see what is happening it check the current 15 milliseconds not the pervious 15 milliseconds and it charges what happens in that 15 milliseconds. So the Activity happens between this time is not accounted. So the threads between is not charged and there are lot of application today that makes thread so fast that it skips from Clock time. So maybe you System might be slow and CPU usage will be like 10% so every time when a thread begins to run it triggers something called Context Switch count which is a heavy operation in Windows. So Process Explorer has created two pseudo Processes called **Interrupts and DPCs** which tracked the count of number of time threads began. The way you can see it in Process Explorer is through "**Context Switch Delta**" column. You can select from the Columns. Under Process Performance Tab.



The advantage is that you could sort this by CSwtich Delta and see when one has the highest number and if that is unnecessary you could kill them and atleast find which process is taking up all the CPU resources.

Interrupt Time Accounting

Interrupt time is nothing but when time the Clock interrupts a Process by a Device driver since the last refresh. Windows does call or switch to a Interrupt handling switch for Interrupt time when a device driver called an Interrupt the just stays the device driver calls an Interrupt then it execute the thread that was interrupted Task Manager doesn't show that Interrupt time or DPC time it shows as Idle time so if you have a high Idle time it might be the an issues with high Interrupt time or DPC time. You can find that by just moving the mouse. When you move the mouse you can see the Interrupt time or DPC count increases.

The System doesn't keep track of which device driver has a high Interrupt time. So if your troubleshooting Interrupt time you could use Microsoft Windows Performance Toolkit designed for analysis of a wide range of performance problems including application start times, boot issues, deferred procedure calls and interrupt activity (DPCs and ISRs), system responsiveness issues, application resource usage, and interrupt storms. Check this links

http://msdn.microsoft.com/en-us/performance/cc752957.aspx

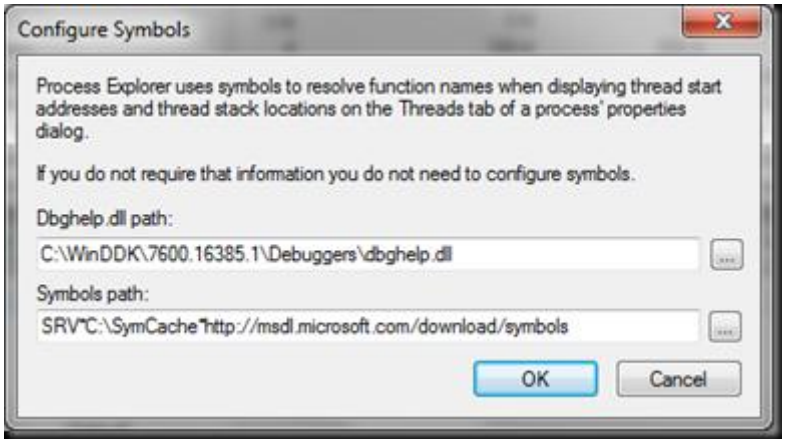
http://msdn.microsoft.com/en-us/library/ff545764%28VS.85%29.aspx

http://msdn.microsoft.com/en-us/magazine/ee358703.aspx

Some Useful Tips:

Configure Process Explorer:

Before getting you start troubleshooting using Process Explorer you have to Configure Symbols.


Configure Symbols
Process Explorer uses symbols to resolve function names when displaying thread start addresses and thread stack locations on the Threads tab of a process' properties dialog.
If you do not require that information you do not need to configure symbols.
Dbghelp.dll path:
C:\WinDDK\7600.16385.1\Debuggers\dbghelp.dll
Symbols path:
SRV*C:\SymCache*http://msdl.microsoft.com/download/symbols
OK
Cancel

· First download Windows Debugging Tools from Microsoft and install it.

· Then Go to Options Menu and Select Configure Symbols

· For Dbghelp.dll path browse to the Debugging Tool install directory and select "**Dbghelp.dll"** File and press OK

· Under Symbol path type the same as you see in the above picture

· The press OK

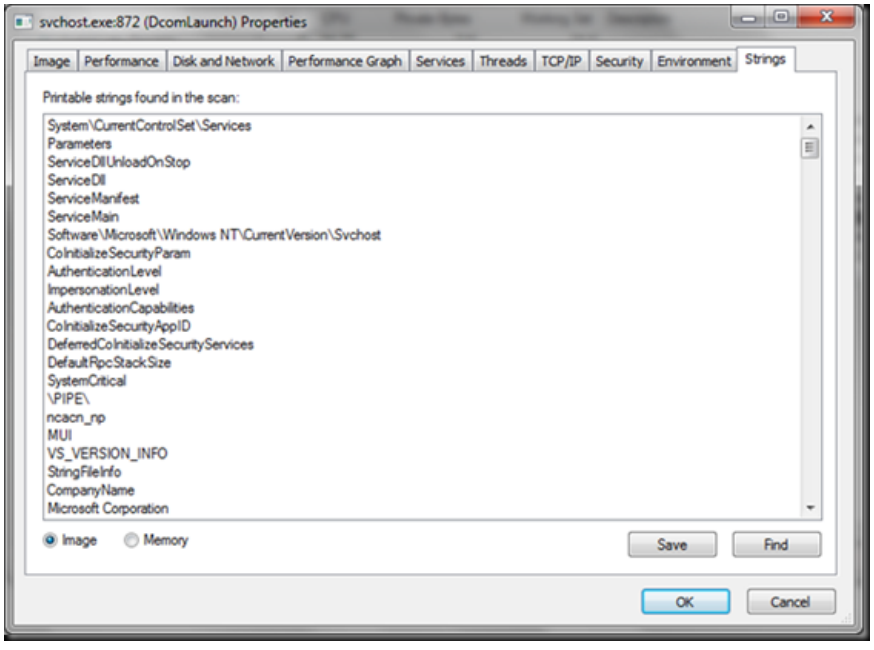
Now you have configure Process Explorer to access the Symbol files if needed. But if you didn't configure it will prompt you select Dbghelp.dll.

Suspend Process:

Suspend process is helpful to not to kill a process but just to pause it. For example if your copying a Large file for a Network and you want to do something important in between you can suspend that Process then do the work and resume it.

This feature also comes handy if you're troubleshooting a Hang process or Memory Leak etc. For example you have a process that is taking around 100% or so CPU or few processes that is taking lo of CPU and you're not sure how to connect between them, just suspend the process so the related process also would stop so now you can connect between them. If one of the System service is consuming CPU and when you suspend a process some other process will stop so we now know which process is causing the System service to consume the CPU.

Strings Explained:



All printable strings of at least 3 characters in length display on this page. Image strings are read from the process image file on disk whereas Memory strings are read from the image's in-memory storage. Memory strings may be different than on-disk strings when an image uses a decompresses or decrypts when it loads into memory.

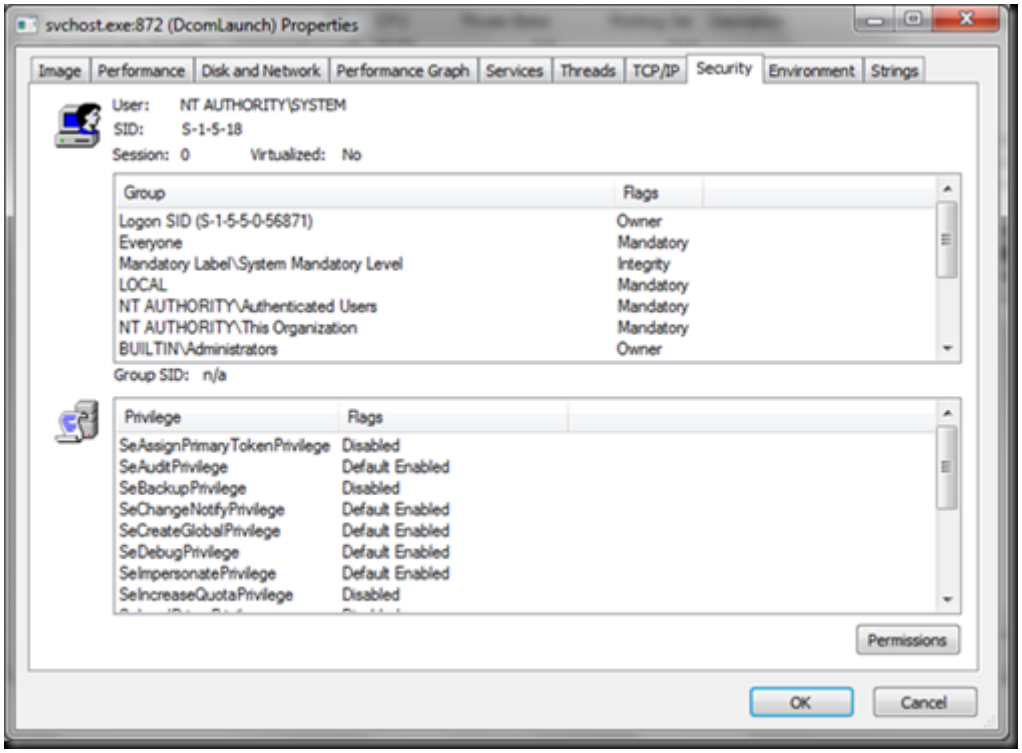
Strings comes helpful if you're trying to find a malicious program. For example you found a process you wanted to know if that will communicate with Web Server or a Website the you could use the Find and search for "www" so it will go through the entire string and show you if it communicates to Internet. If it's a packed process then you can select the Memory button.

Once the process is unpacked to the Memory then it will give you information about that process. Then you can do the same procedure like you do with Image. Most of the malicious software used to be packed images so this would be helpful to find them.

Security Explained:

Process Explorer reports the list of groups and privileges listed in the security token of the process on this page. Privileges shown in grey are disabled. The permissions button opens a permissions editor that shows the access permissions assigned to the process.

This comes helpful when you're troubleshooting security related issues. For example you trying to run a program that needs to copy a file from Desktop to System32 folder. And it failed just says access denied. Even you're the administrator with full admin rights your facing this problem. So you could Right click on that process go to Properties and Click on Security tab.



There you can see the privileges. See if program has that privilege to access that particular folder or to perform that particular operation.

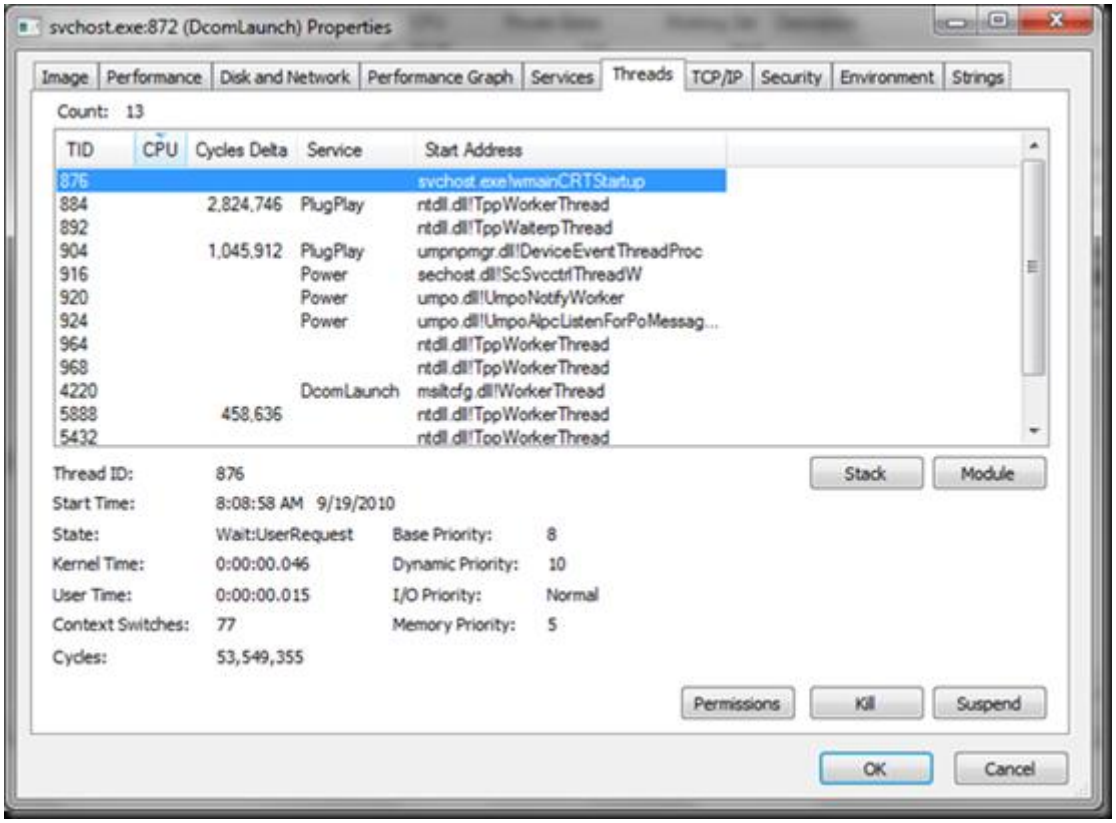
Thread Explained

The list of the threads running in the process shows on this tab. The thread list shows start address information that's provided by the Windows symbol engine. The Module button on the threads page launches Explorer's file properties dialog box for the image file that contains the start address of the currently selected thread. The Stack button shows the current stack of the selected thread. Stack information is unreliable unless symbol files are available for process and DLLs referenced in the stack.

Use the Kill button to terminate a thread. Note that terminating a thread may lead to a crash or erratic behavior of the process.

Use the Suspend button to suspend a thread. Note that suspending threads may cause its process to stop executing.

This comes helpful in troubleshooting hang issues. For example you have an application that takes a long time to load. You could go to the Threads Tab

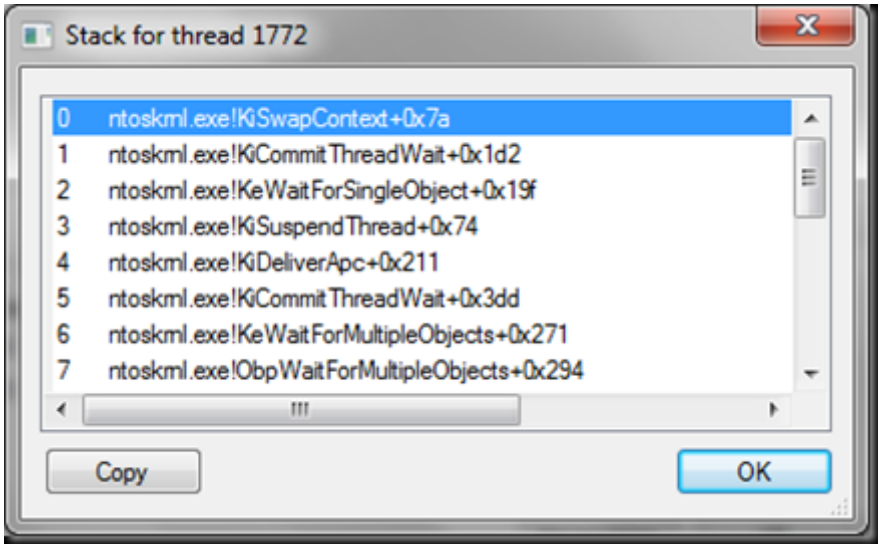


There it will show the threads of that particular process. You can click the Module button to open the Properties of that process. Then Stack button is the important one. Each thread would have its own stack for you for example like **Wait:UserRequest** or **!winspool** etc. then know it's trying to access the printer before opening the application or it's waiting for user's request etc.

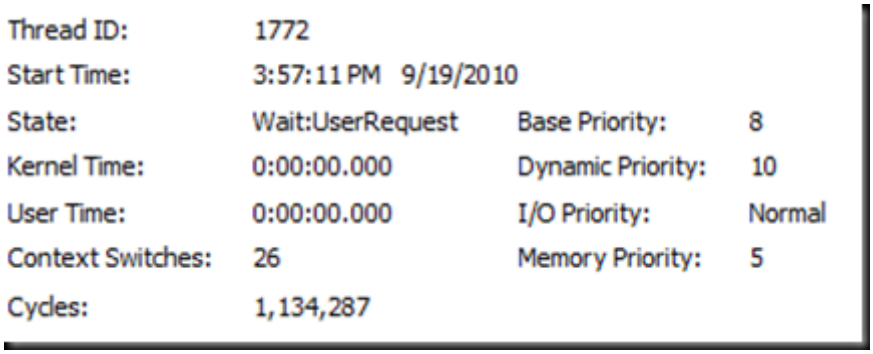
Let's say Microsoft PowerPoint it always tries to access the Printer before opening if the Printer is Offline or if it's a Network printer it might take a while to access it. It has 60 sec of wait time. When you look at the Stack you might see the stack is referring to Printer related service.

If you are not sure but you know something is wrong you click take a snap shot of the Process or create a Dump if that process and send it to the Developer he could answer it. You could use ADPlus Tool that comes with Debugging Tools to create a Memory Snap shot. This Microsoft article explains you how to use it http://support.microsoft.com/kb/286350/en-us. This is a very easy but an expert way of troubleshooting such issues especially when you're troubleshooting a Hung process.

For each thread in the list it shows from the second stack under the Start address section. Because the Start address for all the processes are the same i.e.



For all the Processes you can see the First Stack would be "**ntoskrnl.exe!KiSwapContext +0x7a**" So always it shows from the second Stack under Start address of a thread. As mentioned before make sure you Configure the Symbols properly otherwise trying to troubleshoot this is waste. It won't show you the accurate stacks.



This information sometimes might be useful you can know when did the thread started and when it started at Kernel etc.

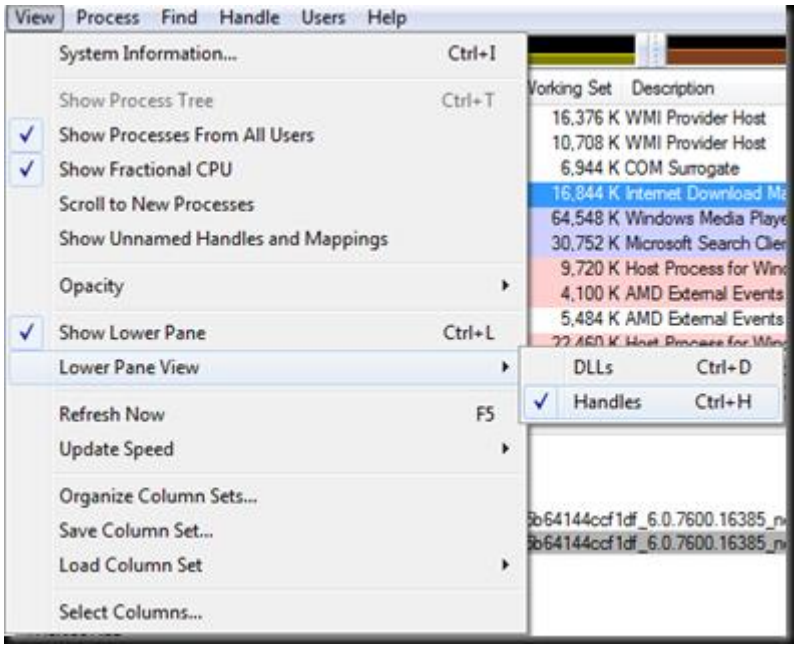
Then like you have in Processes in Thread also you have an option to Suspend and Kill a thread***. It's not recommended*** to do so unless you're sure about that is the thread related to or you might damage the process or whatever task it's running.

Handle Level Troubleshooting

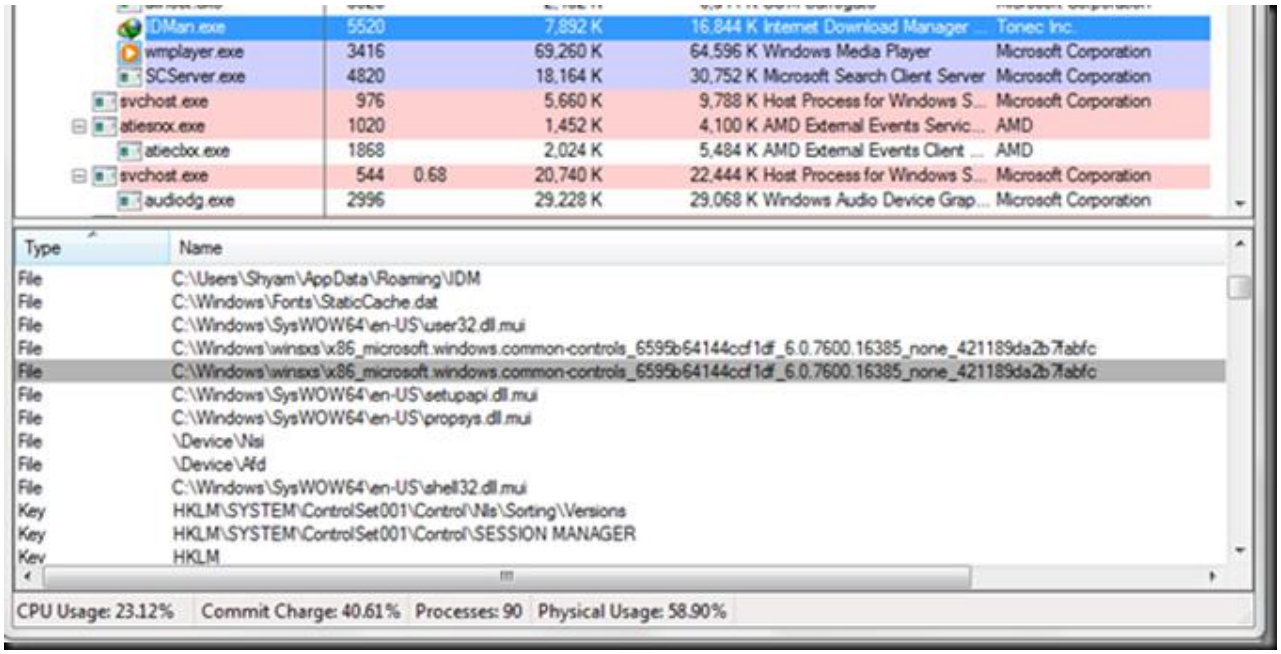
One of the attributes of a process called Handle Table. The Handle table records what Operating System resources are open by threads within that process. Any time a thread opens a resource a handle is created. The handle will continuously inference with that thread

using that handle value. The resources can be Files, Devices, Registry key, TCP/UDP port etc. This will helpful to find the resources open by a process. Process Explorer will show you the Handle table. Process Explorer uses a Driver to do these operations .

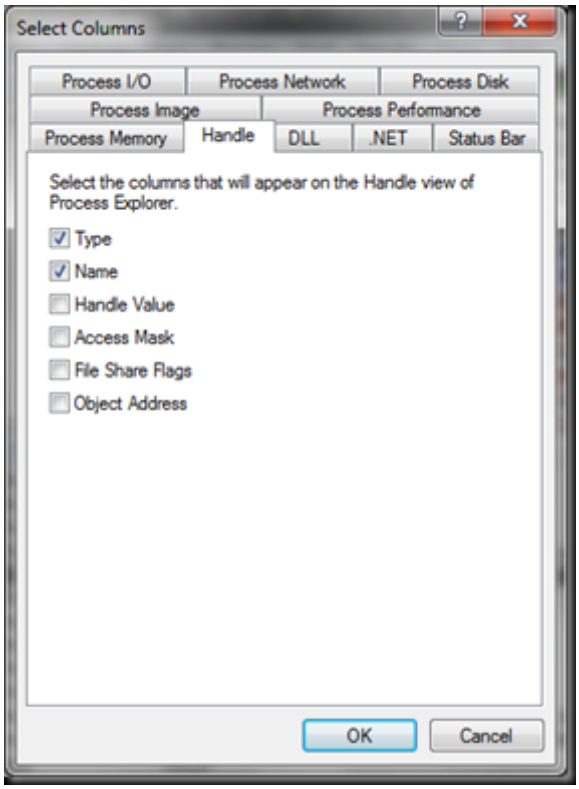
To open the Handle table just open up the lower pane view.



The below picture shows you how it looks like


IDMan.exe 5520 7,892 K 16,844 K Internet Download Manager Tonec Inc.
wmplayer.exe 3416 69,260 K 64,596 K Windows Media Player Microsoft Corporation
SCServer.exe 4820 18,164 K 30,752 K Microsoft Search Client Server Microsoft Corporation
svchost.exe 976 5,660 K 9,788 K Host Process for Windows S... Microsoft Corporation
atiesxx.exe 1020 1,452 K 4,100 K AMD External Events Servic... AMD
atiecbx.exe 1868 2,024 K 5,484 K AMD External Events Client ... AMD
svchost.exe 544 0.68 20,740 K 22,444 K Host Process for Windows S... Microsoft Corporation
audiodg.exe 2996 29,228 K 29,068 K Windows Audio Device Grap... Microsoft Corporation
Type Name
File C:\Users\Shyam\AppData\Roaming\IDM
File C:\Windows\Fonts\StaticCache.dat
File C:\Windows\SysWOW64\en-US\user32.dll.mui
File C:\Windows\winsxs\x86_microsoft.windows.common-controls_6595b64144ccf1df_6.0.7600.16385_none_421189da2b7fabfc
File C:\Windows\winsxs\x86_microsoft.windows.common-controls_6595b64144ccf1df_6.0.7600.16385_none_421189da2b7fabfc
File C:\Windows\SysWOW64\en-US\setupapi.dll.mui
File C:\Windows\SysWOW64\en-US\propsys.dll.mui
File \Device\Nsi
File \Device\Afd
File C:\Windows\SysWOW64\en-US\shell32.dll.mui
Key HKLM\SYSTEM\ControlSet001\Control\Nls\Sorting\Versions
Key HKLM\SYSTEM\ControlSet001\Control\SESSION MANAGER
Key HKLM
CPU Usage: 23.12% Commit Charge: 40.61% Processes: 90 Physical Usage: 58.90%

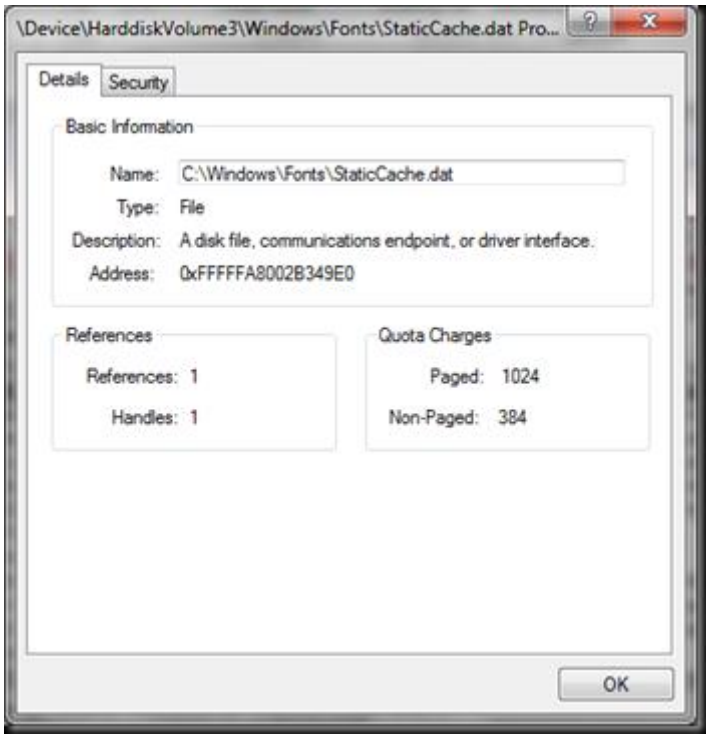
By default it shows you the Type and Name just Right click on the Header and Click on select columns to add additional columns



As If now I'm selecting all the columns for explained a little bit about Handles.

Type	Handle	Object Address	Share Flags	Access
File	0x1F4	0xFFFFFA80030...	RW-	0x00100020
File	0x2AC	0xFFFFFA8002B...	R-D	0x00120089
File	0x2B4	0xFFFFFA80072...	R-D	0x00120089
File	0x2B8	0xFFFFFA8002C...	RW-	0x00100020
File	0x2C4	0xFFFFFA8002F...	RW-	0x00100020
File	0x2D8	0xFFFFFA8002B...	R-D	0x00120089
File	0x310	0xFFFFFA80025...	R-D	0x00120089
File	0x470	0xFFFFFA80074...		0x00100080
File	0x488	0xFFFFFA80025...		0x0012019F
File	0x4B0	0xFFFFFA8002F...	R-D	0x00120089
Key	0x1C	0xFFFFF8A0034...		0x00020019
Key	0x2C	0xFFFFF8A00D...		0x00000001
Key	0x34	0xFFFFF8A00C6...		0x000F003F
Key	0x50	0xFFFFF8A0023...		0x00000001

Under the Handle the values you see "0x1F4" is the values that program uses to Program or call Handle. And the Access shows you what kind of access is granted to that Handle. These information is basically for developers while they debug their application. You can double click on one of the Object to see the Properties. This gives a description of what it does etc.



You also have a search option to search one particular handle. This comes very handy when you're troubleshooting an "Open File" scenario. For example your trying to close a Word document it says it been used be a Program or process. You basically your stuck. So you could use the Find option from Men bar and search for .docx it will list you the location and what program it's using. Also you have an option to close an open handle. So by closing that handle you might be able to delete it. Sometime it might be open in Dll view as well. The processes also load the files to Memory address space or map to the memory for high speed access mechanism.

Also you might have noticed you trying to eject a USB or an External HDD and it fails and says it been used by a Process or Program so you could use the Find option and search for the Drive Letter it will show you which process is using.

This is also useful while you troubleshoot an application. Because it shows you the resources that are open like files, dlls, Registry keys etc. so just looking at it you might find the problem.

Another important and greatly effective use is to find Handle leaks. If your System is extremely slow you have troubleshot all possible ways but a Handle leak can cause the System performance. In Handle view a newly open Handle will be highlighted in green color and closing handle will be in Red. If you see lot of greens and few red you know the Handles are open but they are not closing. So it takes up all the System resources. If you want to look at the number just go to System Information in Process Explorer it show the number of open Handle if that number is keep growing then you have a Handle leak.

Unfortunately there is no easy troubleshooting it fix it but just kill and process and restart. If that didn't fix it then you might want to call their support and inform them there is a handle leak on this process.

Name	Description	Company Name	Version
msvcrt.dll	Windows NT CRT DLL	Microsoft Corporation	7.0.7600.16385
PROPSYS.dll	Microsoft Property System	Microsoft Corporation	7.0.7600.16385
oleacc.dll	Active Accessibility Core Component	Microsoft Corporation	7.0.0.0
oleaccrc.dll	Active Accessibility Resource DLL	Microsoft Corporation	7.0.0.0
MFC42u.dll	MFCDLL Shared Library - Retail Ve...	Microsoft Corporation	6.6.8063.0
MFC42u.dll.mui	MFCDLL Shared Library - Retail Ve...	Microsoft Corporation	6.6.8063.0
COMCTL32.dll	User Experience Controls Library	Microsoft Corporation	6.10.7600.16385
SHELL32.dll	Windows Shell Common Dll	Microsoft Corporation	6.1.7600.16644
OLEAUT32.dll		Microsoft Corporation	6.1.7600.16567
ntdll.dll	NT Layer DLL	Microsoft Corporation	6.1.7600.16559
ADVAPI32.dll	Advanced Windows 32 Base API	Microsoft Corporation	6.1.7600.16385
COMDLG32.dll	Common Dialogs DLL	Microsoft Corporation	6.1.7600.16385
CRYPTBASE.dll	Base cryptographic API DLL	Microsoft Corporation	6.1.7600.16385
CRYPTSP.dll	Cryptographic Service Provider API	Microsoft Corporation	6.1.7600.16385

Like Handle view there is also Dll View that will how you the dlls loaded or opened by that Process. One useful thing about Handle view is to detect version problem of a dlls. If you see a Process is using an older version dll that might be causing trouble to that program

Using Process Explorer to Identify Malicious Software

Process Explorer is a great Tool to help us identify if you have a malicious process running.

Check the Company Name and description of the process: Most of the commercial software will have a Company name and description if your finding something which doesn't then you might research more about that process. This won't be very effective now days because lot of Malware fake them as a Process belongs to Microsoft so it's hard to find if that's the case.

Check the Path: If you see a Windows process that is running from a suspicious location then you might want to double check about that process make sure whether it' s a malware or not.

Check the Strings: Go to Properties of that Process and go to the Strings tab then you could use the Find and search for "**www**" or "**http**" so it will go through the entire string and show you if it communicates to Internet. If it's a packed process then you can select the Memory button. Do the same search

Check the Digital Signature: The Malware can fake them as a Microsoft or a Legit company but they can't fake their digital signature. Use the" **Verify**" option under the Image Tab this function will verify whether it has a valid digital signature or if it says unable to verity. If you find a Process that couldn't verity that's a red alert dig deeper about that process.